Definición de Servicios en RSL para una Infraestructura de Servicios Web de Sistemas de Información Geográfica

Germán Montejano [1 2]; Oscar Testa[2]; Pablo García[2]; Silvia Bast[2]
[1]Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950 – (5700) San Luis – San Luis – Argentina
Tel.: +54-2652-424027 – Int. 251
[gmonte]@unsl.edu.ar – web: http://www.sel.unsl.edu.ar
[2]Departamento de Matemática
Universidad Nacional de La Pampa
Av. Uruguay 151 – (6300) Santa Rosa – La Pampa – Argentina
Tel.: +54-2954-425166– Int. 28
[otesta, pablogarcia, silviabast]@exactas.unlpam.edu.ar

RESUMEN

En este trabajo se realizan aportes a la Ingeniería de Software, más precisamente a la Ingeniería de Software orientada a servicios. Se analizan distintos paradigmas de la orientación a servicios como Service-oriented computing (SOC), Service-oriented programming (SOP), Web Services y Service-oriented architecture (SOA), que es la arquitectura de software sobre la cual se basan los otros paradigmas. Del análisis minucioso de estos conceptos, se realiza una ontología de la cual se deriva un modelo de orientación a servicios que luego se especificará formalmente a través de la utilización de algún lenguaje para tal fin (RSL, MAUDE).

Una vez realizado este trabajo, se validará el modelo propuesto a través de un caso específico: "Una Arquitectura abierta y estándar de servicios Web de GIS, también especificada formalmente en RSL" (Tesis Maestría en Ingeniería de Software).

La necesidad de contar con un modelo de estas características surge debido a la gran diversidad existente de paradigmas y de enfoques acerca de la orientación a servicios, que a su vez permita el desarrollo de sistemas orientados a servicios en forma rápida y sencilla, siguiendo modelos establecidos y probados. Esto permitirá la construcción de sistemas a partir de modelos probados y de utilización directa.

Palabras clave: *RAISE, RSL, Servicios Web, Framework, Infraestructura, Métodos formales, Programación Orientada a Servicios, Computación Orientada a Servicios, Arquitectura Orientada a Servicios.*

CONTEXTO

El presente trabajo se enmarca en el Proyecto de Investigación: Ingeniería de Software, Conceptos, Métodos y Herramientas en un Contexto de "Ingeniería de Software en Evolución" – Facultad de Ciencias Físico-Matemáticas y Naturales, Universidad Nacional de San Luis y en el Proyecto de Investigación: Definición y Especificación Formal de un Modelo basado en Servicios para la Generación de Sistemas de Software – Facultad de Ciencias Exactas y Naturales, Universidad Nacional de La Pampa. Las líneas aquí presentadas fueron expresadas en la Tesis de Posgrado en la Maestría en Ingeniería de Software auspiciada por el proyecto de la Universidad Nacional de San Luis. Actualmente se está trabajando sobre el proyecto de investigación mencionado, donde se espera poder definir formalmente un Modelo basado en Servicios para la generación de Sistemas de Software.

1. INTRODUCCIÓN

La capacidad para responder rápidamente ante los cambios y optimizar los procesos de negocio es un factor clave para la competitividad y el crecimiento de las organizaciones. La agilidad de éstas puede verse cuestionada si se apoya en entornos de IT que no pueden responder de forma flexible a los cambios que afectan a la actividad del negocio. Liberar el potencial que poseen las aplicaciones y recursos de IT y hacerlo disponible de forma general a toda la organización facilita la optimización de procesos y mejora la agilidad empresarial. La Arquitectura Orientada a Servicios (SOA, Service Oriented Architecture) es una filosofía de diseño que permite un mejor alineamiento de las Tecnologías de Información (IT) con las necesidades de negocio[7].

Las empresas necesitan poder interconectar los procesos, personas e información tanto con la propia organización como -atravesando sus fronteras- con subsidiarias y socios comerciales. La falta de integración entre los componentes de IT –sistemas, aplicaciones y datos- hace difícil obtener una respuesta rápida y efectiva ante los cambios que afectan de forma natural a los negocios. La inflexibilidad genera costos, reduce la capacidad de respuesta ante los clientes, compromete el cumplimiento con las normativas legales y afecta negativamente en el personal de la empresa.

En síntesis, una deficiente integración es uno de los problemas más importantes a los que las

organizaciones deben hacer frente para mantener su competitividad y garantizar su crecimiento[7].

El software administrativo está estrechamente unido a la organización interna, procesos y modelos de negocio. Este software subyace tanto en las dependencias interdepartamentales como en las relaciones exteriores de la empresa. En consecuencia, una arquitectura de software administrativo debe hacer frente a un gran número de requisitos diferentes. Muchos de estos requisitos son contradictorios, mientras que otros no están claros. En casi todos los casos, los requisitos son un blanco en movimiento por el cambio permanente de mercados, la organización de la empresa, y sus objetivos de negocio. Es por esto que el desarrollo de software administrativo se hace realmente muy complejo.

Las aplicaciones administrativas de las organizaciones rara vez contienen una gran cantidad de algoritmos complicados. El código que describe un trozo de lógica de negocio suele ser muy simple. La estructura de un sistema de facturación basado en COBOL es mucho más simple que, por ejemplo, un sistema embebido para enviar un robot a Marte (que contiene necesidades de cálculos complejos en tiempo real, y necesidades multi-threading). En las aplicaciones empresariales usualmente se encuentran estructuras de datos muy simples, a diferencia, nuevamente, de otros sistemas tales como sistemas de información geográfica (SIG)[8].
Los sistemas administrativos normalmente tienen que interactuar con distintos stake holders, incluso algunos de ellos pueden ser CEOs de otras compañías, proveedores o usuarios que incluso se ven influenciados por entornos políticos (de la propia empresa o del propio gobierno) que hacen realmente complejo contemplar todas estas variables dentro del sistema a desarrollar.
En cambio en otros tipos de software (software de escritorio, embebido, etc), el entorno y el espacio del problema son mucho más pequeños e incluso más acotado[8].

Se puede concluir que los sistemas o el software administrativo es único en varios aspectos y, por lo tanto, requiere el uso de medidas y metodologías únicas para mejorar su desarrollo y mantenimiento posterior[8].

Como resultado de la estrecha relación con la organización interna, los procesos y modelos de negocio, una arquitectura de software para la empresa debe cumplir con requisitos muy diferentes que los de una arquitectura de software para un sistema que es controlado por un pequeño número de expertos altamente calificados de dominio, como el robot a Marte o un motor de juego de vídeo.

Para poder brindar agilidad y eficiencia, una arquitectura de software administrativo debe contemplar características particulares, a saber: simplicidad, flexibilidad y mantenimiento, reusabilidad y, por último, poder desacoplar la funcionalidad y la tecnología.

A través de la utilización de una Arquitectura Orientada a Servicios (SOA) se puede ayudar a lograr los objetivos de diseño para el desarrollo de software administrativo, planteados anteriormente. Para ello se introducirán algunos conceptos de la orientación a servicios.

El término "servicio" ha estado presente en la computación comercial durante mucho tiempo y ha sido utilizado de muchas maneras diferentes. Hoy, por ejemplo, nos encontramos con grandes empresas, tales como IBM, promocionando el concepto de "servicios a la carta". A principios del nuevo siglo, el término "servicios web" se convirtió en muy popular, aunque a menudo se ha utilizado para referirse a conceptos muy diferentes de computación. Algunas personas lo utilizan para referirse a la aplicación de servicios prestados a los usuarios humanos a través de Internet. Otras personas lo utilizaron para referirse a los módulos de aplicación accesibles para otras aplicaciones a través de Internet, haciendo uso de los protocolos basados en XML.
Debido a las diferentes maneras en que el término “servicio” se ha utilizado en los años de la industria de TI, se hace necesaria la elaboración de una definición más precisa del término[8].

Se puede definir entonces Arquitectura orientada a servicios (SOA) como “una arquitectura débilmente acoplada diseñada para satisfacer las necesidades de negocio de una organización”[9].
SOA es un enfoque arquitectural para la creación de sistemas a partir de servicios autónomos. Esta arquitectura permite la creación de sistemas de software altamente escalables que reflejan el negocio de la organización, brindando además una forma bien definida en la forma en cómo se exponen y se invocan los servicios, lo que facilita la integración e interacción de sistemas tanto propios como de terceros.

La Arquitectura SOA establece un marco de diseño para la integración de aplicaciones independientes de manera que desde la red pueda accederse a sus funcionalidades, las cuales se ofrecen como servicios. La forma más habitual de implementarla es mediante Servicios Web, una tecnología basada en estándares e independiente de la plataforma, con la que SOA puede descomponer aplicaciones monolíticas en un conjunto de servicios e implementar esta funcionalidad en forma modular. En este punto es importante brindar una definición

de “servicio”. En la literatura existen varias definiciones de este término, sin embargo este trabajo utiliza la siguiente conceptualización:

“Un servicio es una funcionalidad concreta que puede ser descubierta en la red y que describe tanto lo que puede hacer como el modo de interactuar con ella.”

Desde la perspectiva de la empresa, un servicio realiza una tarea concreta: puede corresponder a un proceso de negocio tan sencillo como introducir o extraer un dato como “Código del Cliente”. Pero también los servicios pueden acoplarse dentro de una aplicación completa que proporcione servicios de alto nivel, con un grado de complejidad muy superior –por ejemplo, “introducir datos de un pedido”-, un proceso que, desde que comienza hasta que termina, puede involucrar varias aplicaciones de negocio.

La estrategia de orientación a servicios permite la creación de servicios y aplicaciones compuestas que pueden existir con independencia de las tecnologías subyacentes. En lugar de exigir que todos los datos y lógica de negocio residan en un mismo ordenador, el modelo de servicios facilita el acceso y consumo de los recursos de IT a través de la red.

Puesto que los servicios están diseñados para ser independientes, autónomos y para interconectarse adecuadamente, pueden combinarse y recombinarse con suma facilidad en aplicaciones complejas que respondan a las necesidades de cada momento en el seno de una organización. Las aplicaciones compuestas (también llamadas “dinámicas”) son lo que permite a las organizaciones mejorar y automatizar sus procesos manuales, disponer de una visión consistente de sus clientes y socios comerciales, y orquestar sus procesos de negocio para que cumplan con las regulaciones legales y políticas internas.

El resultado final, es que las organizaciones que adoptan la orientación a servicios pueden crear y reutilizar servicios y aplicaciones, adaptarlos ante los cambios evolutivos que se producen dentro y fuera de ellas, y con ello adquirir la agilidad necesaria para ganar ventaja competitiva.

La figura 1 muestra un resumen de cómo se puede descomponer un proceso en servicios.

2. LÍNEAS DE INVESTIGACIÓN y DESARROLLO

Tal como se menciona en la Introducción, la capacidad para responder rápidamente ante los cambios y optimizar los procesos de negocio es un factor clave para la competitividad y el crecimiento de las organizaciones. La agilidad de éstas puede verse cuestionada si se apoya en entornos de IT que no puedan responder de forma flexible a los cambios que afectan a la actividad de negocio.

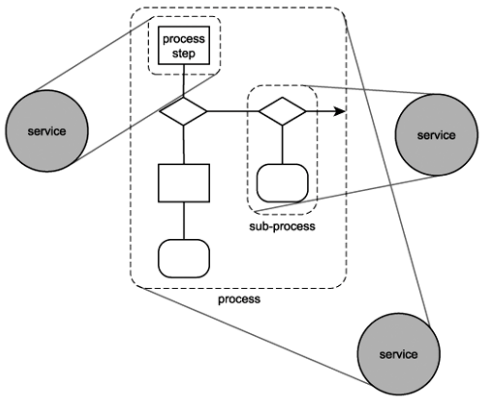

Figura 1. Descomposición de un proceso en servicios.

Con el ánimo de dar respuesta a las necesidades planteadas anteriormente, se genera la arquitectura orientada a servicios, que permite enfocar la solución de los problemas de desarrollo de software, para empresas de una manera más sencilla, rápida y ágil. De esta forma es posible conseguir una adaptación fácil a los cambios organizacionales y estructurales de la empresa. A partir de esta arquitectura, también se han generado nuevos paradigmas de programación, como por ejemplo: Programación orientada a servicios (SOP), que utiliza los servicios como una unidad de trabajo computacional, para diseñar e implementar aplicaciones de software integradas de negocio.

Otro de los paradigmas que emergió es el de Computación Orientada a Servicios (SOC), un enfoque para proveer flexibilidad y agilidad, no solamente en el desarrollo de sistemas, sino también en los procesos de negocios. También se debe recordar que la orientación a servicios se basa en metodologías y paradigmas existentes como los Servicios Web, la Programación orientada a objetos, programación orientada a componentes. SOA tiene sus bases también en la Arquitectura basada en modelos (MDA), un enfoque de diseño de software para el desarrollo de aplicaciones.

El éxito de la implementación en las organizaciones de un sistema de software se basa fundamentalmente en la correcta utilización de las tecnologías, metodologías y paradigmas enunciados anteriormente, que son las que le van a permitir a la organización desarrollar sistemas que se puedan, no sólo implementar, sino también mantener y adecuar a las necesidades cambiantes de la institución y del entorno en el cual se desempeña.

Se tiene la fuerte convicción de que si las organizaciones tuvieran herramientas que les permitieran desarrollar sistemas en forma rápida y

sencilla, siguiendo modelos / técnicas / arquitecturas / paradigmas / metodologías establecidos, podrían tener ventajas competitivas respecto del resto.

Debido a la diversidad de metodologías y modelos existentes para poder desarrollar sistemas de software integrados a los negocios de las empresas, con una visión orientada a servicios es que se propone la construcción, a través de métodos formales, de un modelo de desarrollo de software orientado a servicios, que permita la construcción de sistemas de software a partir de modelos probados y de utilización directa.

La hipótesis más importante que se plantea, por lo tanto, es que si las organizaciones contaran con herramientas que les permitan desarrollar sistemas en forma rápida y sencilla siguiendo modelos establecidos, probados y de utilización directa tendrían ventajas competitivas respecto del resto. Esto permitiría un mejor posicionamiento en el mercado frente a los desafíos que se presentan hoy en día.

Cuando se menciona la construcción de un modelo de desarrollo, no significa estrictamente un modelo nuevo, sino por el contrario, se intentará poder tomar la mejor estrategia luego de hacer una exhaustivo análisis de las tecnologías existentes. Luego de ello se decidirá si es mejor o no realizar una construcción nueva, mejorar alguna existente, o bien tomar la mejor alternativa ya probada.

Las líneas de investigación se basarán principalmente en los siguientes tópicos, a saber:

- Búsqueda minuciosa acerca de las tecnologías / metodologías en relación al ámbito de este proyecto de investigación. Este análisis permitirá realizar un cuadro comparativo que permitirá evaluar las fortalezas y desventajas de cada una de las tecnologías. De este análisis de fortalezas y desventajas se podrá tomar una resolución de las tecnologías, lo que permitirá obtener conclusiones acerca de cómo debería ser un modelo teórico para el desarrollo de sistemas enfocados a servicios.
- Definición de manera teórica del modelo a utilizar y luego especificar formalmente para el desarrollo de sistemas orientados a servicios. La definición del modelo se realizará de manera informal, es decir en un lenguaje ambiguo, pero que sentará las bases para luego poder especificarlo formalmente. En esta definición del modelo se incluirán todos los detalles necesarios para que el pasaje a lenguajes formales sea lo más directa posible. Para la definición del modelo se utilizarán todas las herramientas que puedan facilitar la tarea, como, por ejemplo, diagramas UML.
- Formalización del modelo definido durante el segundo año de trabajo. Para esta tarea se utilizarán lenguajes de especificación formal como son RSL.

El lenguaje de especificación formal utilizado en este caso es el RAISE Specification Language (RSL), dado que el mismo es reconocido en la industria del software para especificaciones formales de desarrollos reales.

RSL es un lenguaje formal, basado en el formalismo de la matemática usando conceptos tales como la teoría de conjuntos, lógica de primer orden, lógica de orden superior. Está netamente orientado a construir modelos, ya sea describiendo un dominio de la realidad o describiendo una herramienta a desarrollar y sus requerimientos

Durante el transcurso de los dos primeros años también se harán estudios del estado del arte respecto de lenguajes para especificaciones formales, pudiendo surgir otra alternativa de lenguaje a utilizar como puede ser MAUDE o Alloy (sólo por mencionar algunos). Si bien se asume que se utilizará RSL, se espera evaluar otras alternativas.

3. RESULTADOS OBTENIDOS/ESPERADOS

Se evidencia entonces que de existir un modelo con esas características, más aún si se encuentra basado en especificaciones formales, el desarrollo de aplicaciones de software basada en servicios se vería beneficiado ampliamente tanto en el ámbito privado como en el de organismos públicos. Luego se validará y verificará el modelo especificado y propuesto a través de la utilización en un caso concreto, Framework para Sistemas de Información Geográfica.

De acuerdo a lo planteado se desprende que existe un campo de trabajo interesante (y con áreas no cubiertas) en el ámbito del problema, y más aún en su especificación formal.

4. FORMACIÓN DE RECURSOS HUMANOS

Además de los resultados obtenidos/esperados en el punto 3, se espera como resultado en la formación de recursos humanos, la continuación de esta misma línea de proyecto como tesis doctoral de alguno(s) de los investigadores así como también la mayor interrelación con la Universidad de Minas Geráis con la que contamos con un convenio con tal objetivo como parte de él. Adicionalmente, se espera que otras tesis de Maestría, así como tesinas de Licenciatura surjan a partir de los logros obtenidos en la presente línea investigativa.

5. BIBLIOGRAFÍA

1. The RAISE Method Group, “The RAISE Development Method”, The Practitioner Series, PrenticeHall, 1995.
2. Clarke, E; Wing, J., “Formal Methods: State of the Art and Future Directions”.
3. Ethan, Cerami, “Web Services Essentials – Distributed Applications with XML-RPC, SOAP, UDDI & WSDL”, O’Reilly, 1th ed, 2002.
4. Newcomer, Eric, “Understanding Web Services Xml Wsdl Soap And Uddi”, Addison Wesley.
5. The RAISE Language Group, “The RAISE Specification Language”, The BCS Practitioner Series, PrenticeHall, 1992.
6. Dang, Van Hung, George, Chris, Janowski, Tomasz, Moore, Richard, “Specification Case Studies in RAISE”, UNU-IIST, 2002.
7. Microsoft Corporation, “La Arquitectura Orientada a Servicios (SOA) de Microsoft aplicada al mundo real”, Whitepaper, Diciembre de 2006.
8. Dirk, Krafzig; Karl, Banke; Dirk, Slama, “Enterprise SOA: Service-Oriented Architecture Best Practices”, Prentice Hall PTR, 2004.
9. James P. Lawler; H. Howell-Barber, “Service-Oriented Architecture. SOA Strategy, and Technology”, Auerbach Publications, 2008
10. Thomas Erl, “Service-Oriented Architecture: Concepts, Technology, and Design”, Prentice Hall PTR, 2005.
11. Testa, Oscar; Montejano, Germán, Riesco, Daniel, “Especificación Formal en RSL de una Infraestructura abierta y estándar de Servicios Web de SIG.”, Tesis de Maestría de Ingeniería de Software, Universidad Nacional de San Luis, 2010.